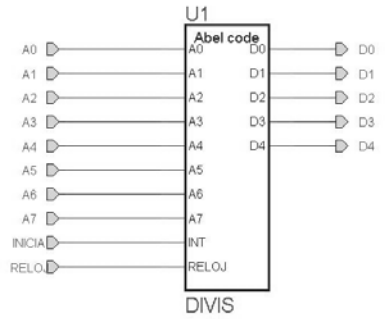
Unidad de División en ABEL HDL.

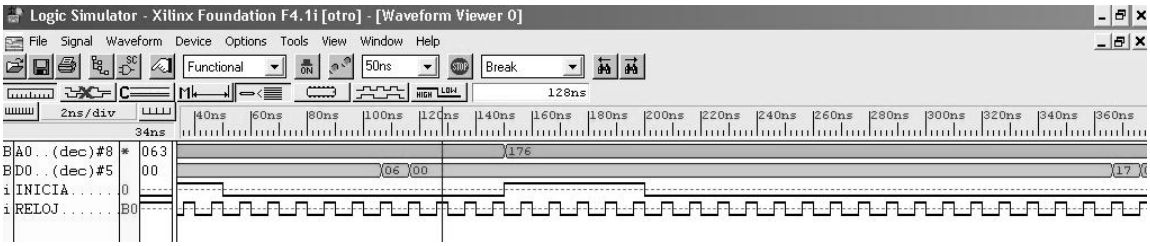
M. en C. Juan Carlos Herrera Lozada, CIDETEC IPN, 2007. jlozada@ipn.mx

En mi caso utilicé el algoritmo de restas sucesivas actualizando un dato temporal que es en sí las veces que se resta el valor constante hasta que el resultado presente un valor menor que el mismo 10.
Observa que utilizo una variable adicional *int* que cuando le asigno un nivel lógico alto, me permite almacenar el dato que se dividirá, posteriormente le asigno a un nivel lógico bajo para comenzar la división; es similar a un *load*.

```
MODULE divis
title 'Divisor por 10'
reloj, a7, a6, a5, a4, a3, a2, a1, a0, int pin ;
d4, d3, d2, d1, d0 pin istype 'com';

"Las salidas del contador se declaran como nodos
di4, di3, di2, di1, di0 node istype 'reg';
divi= [di4, di3, di2, di1, di0];

Count7..Count0 node istype 'reg';
Counter = [Count7..Count0];
div = [d4, d3, d2, d1, d0];
dat = [a7, a6, a5, a4, a3, a2, a1, a0];

Equations
@carry 1;
Counter.clk = reloj;
divi.clk = reloj;

when int then {Counter:=dat; divi:=0}
   else when Counter<10 then {Counter:=Counter; div=divi}
     else {Counter:=Counter-10; divi:=divi+1}
END
```



En la simulación es posible advertir que requiero *int* a 1 lógico para cargar el dato. Cuando *int* es 0, comienzan las restas sucesivas. El resultado es el número de restas realizadas, equivalentes a pulsos de reloj.



El código anterior no resuelve cuando el número a dividir es menor que 10. Para adicionar funcionalidad (altamente recomendado), puedes incluir una lógica que calcule el residuo que te elimina la parte entera del resultado.
¿Qué tan complejo resultaría un código que dividiera dos números de 8 bits (ninguno de los dos es constante)?